LISTING OF THE CLAIMS:

The following is a complete listing of all the claims in the application, with an indication of the status of each:

1. (Currently Amended) A computer-based software task management system comprising:

a Task ID register for storing a plurality of different Task IDs;

a plurality of ~~an~~ index registers, each associated with a corresponding one of the Task IDs, each index register configured to store a data register pointer for pointing to a data register;

~~a Task ID register coupled to the index register and configured to store a Task ID keyed to the index register;~~

a Task ID memory coupled to the Task ID register and configured to store a plurality of flags, each flag associated with a corresponding one of the Task IDs and each indicating whether the corresponding Task ID is available; and

a state machine coupled to the Task ID memory and configured to (a) receive a Task ID request from a task, (b) to determine whether a Task ID is available in response to the Task ID request, (c) when a Task ID is available, to issue a Task ID to the task and set the flag in the Task ID memory indicating that the Task ID is in use, and (d) when the task is

complete, to reset the flag in the Task ID memory indicating that the Task ID is available.

2. (Currently Amended) The computer-based software task management system of claim 1, ~~further comprising:~~

~~a plurality of index registers each configured to store a data register pointer for pointing to a data register~~

wherein the Task ID register comprises a plurality of Task ID registers, ~~each coupled to the index register and~~ each configured to store one of said [[a]] Task IDs, ~~keyed to a respective index register;~~

wherein the Task ID memory comprises a plurality of Task ID memories, each storing one of said plurality of flags ~~each coupled to the Task ID register and configured to store a flag indicating whether a respective Task ID is available~~ [[;]] and

wherein the state machine is configured to manage a plurality of tasks with the plurality of index registers, Task ID registers and the Task ID memory.

3. (Original) The computer-based software task management system of claim 2, wherein each index register is uniquely associated with a different Task ID.

4. (Currently Amended) The computer-based software task management system of claim 2, further comprising:

a flip-flop circuit coupled to each ~~the~~ index register and configured to toggle in response to each instance of the task, to cause the task to alternate between a write cycle to the index register and one selected from the group consisting of: a write cycle to the data register pointed to by the index register; and a read cycle to the data register pointer to by the index register.

5. (Currently Amended) A computer-implemented method for managing multiple tasks using an index register, comprising:

(a) receiving a Task ID request from a task;

(b) reading a Task ID memory in response to the Task ID request to determine ~~determining~~ whether a Task ID is available for ~~in response to~~ the Task ID request;

(c) when a Task ID is available, issuing a Task ID to the task and setting a flag in the [[a]] Task ID memory indicating that the Task ID is in use; [[,]]and

(d) when the task is complete, resetting the flag in the Task ID memory indicating that the Task ID is available.

6. (Currently Amended) The computer-implemented method of claim 5 using a plurality of index registers with a Task ID associated with each index register, wherein: the determining step includes the step of determining whether a Task ID is available from the plurality of Task IDs in response to the Task ID request.

7. (Currently Amended) The computer-implemented method of claim 5, further comprising: when a Task ID is not available, periodically requesting a Task ID.

8. (Currently Amended) The computer-implemented method of claim 5, further comprising the step of: an alternating read/write toggle in response to each instance of the task, causing the task to alternate between a write cycle to the index register and one selected from the group consisting of: a write cycle to the data register pointed to by the index register; and a read cycle to the data register pointer to by the index register.

9. (Currently Amended) The computer-implemented method of claim 6, further comprising the step of: toggling a read/write control flip-flop in response to each instance of the task, causing the task to alternate between a write cycle to the index register and one selected from the group consisting of: a write

cycle to the data register pointed to by the index register; and a read cycle to the data register pointer to by the index register.

10. (Currently Amended) The computer-implemented method of claim 8, further comprising: resetting the read/write [[a]] flip-flop ~~circuit~~ after each read-cycle, the flip-flop ~~circuit~~ steering the read and write accesses to the index register and the data register pointed to by the index register.

11. (Currently Amended) The computer-implemented method of claim 9, further comprising: resetting the read/write [[a]] flip-flop ~~circuit~~ after each read-cycle, the read/write flip-flop ~~circuit~~ steering the read and write accesses to the index register and the data register pointed to by the index register.